APPLICATION FOR UNITED STATES PATENT

by

MARC A. NORTON

and

DANIEL J. ROELKER

for

SYSTEMS AND METHODS FOR DYNAMIC THREAT ASSESSMENT

SHAW PITTMAN LLP
1650 Tysons Blvd., 14th Floor
McLean, Virginia 22102-4859
(703) 770-7900

Attorney Docket No.: SFR0003-US

SYSTEMS AND METHODS FOR DYNAMIC THREAT ASSESSMENT

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0001] Embodiments of the present invention relate to methods and systems for dynamic threat assessment using computer or computer network security devices. More particularly, embodiments of the present invention relate to systems and methods for combining attack identification and event correlation from one or more security devices that generate events. These systems and methods provide a real-time assessment of which internal computers are at future risk for what type of attack, and which external computers may need to be watched more closely in the future.

BACKGROUND INFORMATION

[0002] Most successful computer intrusions and computer network attacks consist of a series of calculated steps. While each step may not normally constitute an intrusion or attack, the culmination of steps often does. In addition, the end goals of each series of steps that constitutes an intrusion or attack vary widely. For example, an attacker may have the goal of compromising a host for conducting future attacks. Such an attacker will most likely do port reconnaissance, host service identification, service exploitation, and finally installation of a rootkit or backdoor. Another attacker may have the goal of compromising a host to take specific information located on that host. Such an attacker may guess a user's password and transfer the desired files back across the network to a zombie host to defeat tracebacks.

[0003] In view of the foregoing, it can be appreciated that a substantial need exists for systems and methods that can advantageously predict or discover an intrusion or attack by analyzing a series of security device events over time.

BRIEF SUMMARY OF THE INVENTION

[0004] Embodiments of the present invention relate to systems and methods for combining attack identification and event correlation information from security devices that generate events. One embodiment of the present invention is a method for assessing a threat probability of events generated by a security device. In this method, an event from a security device is received by an event collection database. If the event matches an unpopulated member of an instance of an abstract data type, the event is added to the instance and the probability of the instance is computed. The instance of an abstract data type represents a rule that describes how events are combined to form a threat. If the probability of the instance is greater than a global threat assessment event generation probability, a second event is generated and the second event is placed in the event collection database.

[0005] Another embodiment of the present invention is a method for dynamically assessing threats to computers and computer networks using security devices that generate events. In this method, policy configuration information is read. This information includes a global threat assessment event generation probability and dynamic threat assessment rules containing event probability information. Abstract data types are generated for each dynamic threat assessment rule. Events from the security devices are collected and stored in an event collection database. An event in the event collection database is read. It is determined if the event is a member of an instance of an abstract data type. If the event is a member of the instance, the event is

added to an instance and the probability of the instance is computed. It is determined if the probability of the instance is greater than the global threat assessment event generation probability. If the probability of the instance is greater than the global threat assessment event generation probability, a dynamic threat assessment event is generated and the dynamic threat assessment event is placed in the event collection database. It is then determined if the event is a starting member of an abstract data type. If the event is a starting member of the abstract data type, an instance of the abstract data type is created and the event is added to the instance.

[0006] Another embodiment of the present invention is a system for dynamically assessing threats to computers and computer networks. The system includes at least one security device that generates events, an event collection database, policy configuration information, and a dynamic threat assessment engine. The event collection database receives and stores events generated by security devices. The policy configuration information includes a global threat assessment event generation probability and dynamic threat assessment rules containing event probability information. The dynamic threat assessment engine reads the policy configuration information and creates abstract data types based on the dynamic threat assessment rules. The dynamic threat assessment engine populates instances of the abstract data types with events from the event collection database. The dynamic threat assessment engine generates events for instances with probabilities greater than the global threat assessment event generation probability and places these events back into the event collection database.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] Figure 1 is a flowchart showing the steps of an exemplary vulnerability that can be detected and ameliorated by an embodiment of the present invention.

[0008] Figure 2 is a flowchart showing the steps of an exemplary method for dynamic threat assessment in accordance with an embodiment of the present invention.

[0009] Figure 3 is a schematic diagram showing an exemplary dynamic threat assessment system in accordance with an embodiment of the present invention.

[0010] Figure 4 is a flowchart showing the steps of an exemplary method for assessing a threat probability of an event generated by a security device in accordance with an embodiment of the present invention.

[0011] Before one or more embodiments of the invention are described in detail, one skilled in the art will appreciate that the invention is not limited in its application to the details of construction, the arrangements of components, and the arrangement of steps set forth in the following detailed description or illustrated in the drawings. The invention is capable of other embodiments and of being practiced or being carried out in various ways. Also, it is to be understood that the phraseology and terminology used herein is for the purpose of description and should not be regarded as limiting.

DETAILED DESCRIPTION OF THE INVENTION

[0012] Embodiments of systems and methods related to combining attack identification and event correlation information from security devices that generate events are described in this detailed description of the invention. In this detailed description, for purposes of explanation, numerous specific details are set forth to provide a thorough understanding of embodiments of the present invention. One

skilled in the art will appreciate, however, that embodiments of the present invention may be practiced without these specific details. In other instances, structures and devices are shown in block diagram form. Furthermore, one skilled in the art can readily appreciate that the specific sequences in which methods are presented and performed are illustrative and it is contemplated that the sequences can be varied and still remain within the spirit and scope of embodiments of the present invention.

[0013] A computer security event or computer network security event is defined herein as logged information associated with an unwanted intrusion or a potentially unwanted intrusion. These events are generated through a process called attack identification. Attack identification in intrusion detection systems (IDSs), for example, involves searching individual packets for patterns indicative of an intrusion or potential intrusion. When a known intrusion pattern is found by an IDS, the IDS generates an event.

[0014] Unfortunately, however, an attack on a computer or computer network may consist of two or more steps or intrusions, which individually would otherwise be innocuous. Once detected, these steps or intrusions result in one or more events. Current IDS technologies enable analysts to detect such events, and some correlation technologies can put multiple events together to give an analyst a view of a "super" event. These technologies are collectively called event correlation.

[0015] One embodiment of the present invention takes attack identification and event correlation one step further. This embodiment analyzes events from a variety of security devices that generate events and determines what combinations of attacks coming from and going to various hosts might indicate that a larger, coordinated

attack is in progress. Security devices that generate events include but are not limited to network intrusion detection systems (NIDs), host intrusion detection systems (HIDs), routers, firewalls, and system loggers.

[0016] This embodiment of the present invention provides a security analyst with a real-time assessment of which hosts are at future risk for what type of attack and which external hosts may need to be watched more closely in the future. This assessment is referred to herein as dynamic threat assessment (DTA). The information provided by this assessment gives the analyst a chance to take immediate measures in preventing further exploitation.

[0017] A DTA system is a dynamic system in that it solves the problem of the "all or nothing" or static approach to intrusion detection and prevention. Currently, intrusion detection technologies are either active or passive, and there is no in between. A security analyst decides which attacks to stop at the gateway and which attacks will be let through and monitored passively. To an analyst, there is a gap between these technologies. That gap is the ability to predict impending attacks, which would allow the analyst to take immediate action on threats that have a high probability of occurring.

[0018] A DTA system solves this problem by allowing probabilities to be assigned and reassigned to one or more events based on a variety of conditions. These conditions include but are not limited to the number of other events, the type of other events, the order of other events, and the timing of other events. The security analyst can then predict and interdict attacks based on an overall probability of attack for the computer or computer network.

[0019] The present invention is based on current methods of hacker exploitation and maps out the various paths that attackers can take to reach their goals. Of course, as one of skill in the art would appreciate, the present invention could be adapted to new methods of hacker exploitation as they develop. Nonetheless, the general approach to DTA is two-fold. The first part is programmable knowledge of how current network and host attacks are defined. This is accomplished through either a dynamic information base, *e.g.*, rules and signatures, or through hard-coding the information base in the technology.

[0020] The second part is accomplished through the integration of abstract data types with the information base in part one. These abstract data types provide the technology for modeling and predicting attack vectors under various degrees of uncertainty. One exemplary method of modeling attack vectors is through the use of an attack tree. An attack tree is a tree structure where the root node represents the attack goal and each leaf node represents a method of achieving that goal. An exemplary method of using attack trees was described by Bruce Schneier in "Attack Trees", Dr. Dobb's Journal, Dec. 1999, and is incorporated herein by reference in its entirety.

[0021] Figure 1 is a flowchart showing the steps of an exemplary vulnerability that can be detected and ameliorated by an embodiment of the present invention. This exemplary vulnerability has been found in the ProFTPd program.

[0022] In step 110 of vulnerability 100, the attacker logs on to an FTP server a first time.

[0023] In step 115, the attacker uploads an attack file to the FTP server.

[0024] In step 120, the attacker logs off of the FTP server a first time.

[0025] In step 125, the attacker logs back on to the same FTP server a second time.

[0026] In step 130, the attacker sets the transfer mode to ASCII.

[0027] In step 135, the attacker downloads the uploaded attack file a first time.

[0028] In step 140, the attacker logs off of the FTP server a second time.

[0029] In step 145, the attacker logs back on to the same FTP server a third time.

[0030] In step 150, the attacker sets the transfer mode to ASCII.

[0031] In step 155, the attacker downloads the uploaded attack file a second time, which triggers a buffer overflow.

[0032] According to one embodiment of a method for DTA of the present invention, an analyst first sets a probability level on the vulnerability 100 of Figure 1 that triggers when a certain number of the steps have been completed. If the probability level is set high (*i.e.*, greater number of steps completed), there is a smaller chance of an invalid assessment. Next, the attacker's IP address is blocked at the firewall when the specified number of steps have occurred and the set probability level of the detection system is less than or equal to the probability of vulnerability 100. In another embodiment, a countermeasure other than blocking the attacker's IP address at the firewall is taken to prevent vulnerability 100 before it occurs.

[0033] Figure 2 is a flowchart showing the steps of an exemplary method for DTA in accordance with an embodiment of the present invention.

[0034] In step 210 of method 200, policy configuration information is read. This information includes a global threat assessment event generation probability and one or more DTA rules. The global threat assessment event generation probability is set

by the user or security analyst and represents the overall probability threshold for generating DTA events.

[0035] DTA rules are the directives that specify how one or more events constitute an attack or threat. These directives include information on how the probabilities of the one or more events that constitute an attack should be computed.

[0036] In another embodiment of this method, the policy configuration information includes rule probability thresholds for each DTA rule. These thresholds provide specific probability limitations for each instance of an abstract data type created from a DTA rule.

[0037] In another embodiment of this method, the policy configuration information includes event collection database configurations. Exemplary configurations include information about how events are collected, received, and stored from one or more security devices and from the DTA system itself. This information includes the ordering and timing of event collection.

[0038] In another embodiment of this method, the policy configuration information includes operation parameters. Exemplary operation parameters include limitations on system memory and CPU usage.

[0039] In step 220, one or more abstract data types are generated for each of the one or more DTA rules. These abstract data types include but are not limited to graphs, trees, lists, state machines, hash tables, and Bayesian networks. Each abstract data type includes probability information obtained from its rule that is used when computing the probabilities of instances of each abstract data type.

[0040] In step 230, events are collected from one or more security devices and are stored in an event collection database. In an alternative embodiment, event information is pushed from the one or more security devices into the event collection database, rather than being pulled from the one or more security devices. In other words, events are received by the event collection database from the one or more security devices.

[0041] In step 240, an attempt is made to read an event from the event collection database.

[0042] In step 250, it is determined if there are any unprocessed events in the event collection database. If there are no more unprocessed events, then collection of events from the one or more security devices (step 230) is attempted again. If there are more unprocessed events, step 260 is executed with the next unprocessed event.

[0043] In step 260, it is determined if the event is a member of any of the current instances of abstract data types. An instance is defined herein as a copy of a abstract data type that can contain actual data. An instance of an abstract data type is created when an event is found that is a starting member of the abstract data type. For example, if the abstract data type is a linked list of three events where the order is important, an instance of the linked list will be created when the event representing the first link, or first member, in the list is found. An instance of an abstract data type remains in memory until the calculated probability of the instance exceeds a probability threshold. In another embodiment of this method, an instance is removed from memory after a certain time period. In step 260, the event is processed against all instances of abstract data types currently in memory. Processing the event against

an instance includes but is not limited to reading lookup information in the event including host IP addresses and comparing that lookup information to a member of the instance. For each instance of abstract data types that includes the event as a member, step 265 is executed. If the event is not a member of any of the instances of abstract data types currently in memory, step 265 is not executed.

[0044] In step 265, the event is added to that instance and the probability of the instance is recomputed. The probability of the instance is recomputed based on one or more conditions relative to the added event. These conditions include but are not limited to the number of other events, the type of other events, the order of other events, and the timing of the other events. For example, consider three different types of events A, B, and C. By themselves, events A, B, and C have threat probabilities of ten, twenty, and thirty, respectively. However, in combination, event A and event B have a threat probability of fifty, and, in combination, event A and event C have a threat probability of zero. Suppose then that event A is the only event of an instance. That instance will have a probability of ten. If event B is added to that instance, however, its probably is recomputed and becomes fifty. If alternatively event C is added to that instance, its probably is recomputed and becomes zero. In this example, the probability of the instance is recomputed based on the type of the event added to the instance, event B or event C, relative to the type of the other event already part of the instance, in this case event A.

[0045] In step 270, the probabilities of all the instances to which the event has been added are compared to the global threat assessment event generation probability. If the probability of an instance is greater than the global threat assessment event

generation probability, step 275 is executed. If the probability of the instance is not greater than the global threat assessment event generation probability, step 275 is not executed. In another embodiment of this method, step 275 is executed if the probability of the instance is greater than or equal to the global threat assessment event generation probability and step 275 is not executed if the probability of the instance is less than the global threat assessment event generation probability.

[0046] In another embodiment of this method, each of the probabilities of all the instances to which the event has been added is compared to its specific rule probability threshold in addition to the global threat assessment event generation probability. If the probability of an instance is greater than its rule probability threshold, step 275 is executed. If the probability of the instance is not greater than its rule probability threshold, step 275 is not executed. In another embodiment of this method, step 275 is executed if the probability of the instance is greater than or equal to its rule probability threshold and step 275 is not executed if the probability of the instance is less than its rule probability threshold.

[0047] In step 275, a DTA event is generated for the instance and is placed in the event collection database. In another embodiment of this method, the DTA event it treated as any other event in the event collection database and is processed in a feedback loop beginning at step 240. In another embodiment of this method, the instance is removed from memory after a DTA event has been generated.

[0048] In step 280, it is determined if the event is a starting member of any of the abstract data types. The event is processed against each abstract data types to determine if a new instance should be created. If the event is a starting member of an

abstract data type, step 285 is executed. If the event is not a starting member of an abstract data type, step 285 is not executed.

[0049] In step 285, an instance of the abstract data type is created and the event is added to the instance. In another embodiment of this method, the probability of the instance is computed and the method continues at step 270 after step 285 has been executed. This allows DTA events to be generated from instances containing a single event.

[0050] When an event has been processed against all current instances of abstract data types and all abstract data types without an instance, the process returns to step 240 to obtain the next unprocessed event.

[0051] Figure 3 is a schematic diagram showing an exemplary DTA system in accordance with an embodiment of the present invention.

[0052] One or more security devices 310 of system 300 perform attack identification and generate events. One or more security devices 310 include but are not limited to IDSs, NIDs, HIDs, routers, firewalls, and system loggers.

[0053] The events generated by one or more security devices 310 are collected and stored by event collection database 320. In another embodiment of this method, event collection database 320 receives and stores events from one or more security devices 310 and DTA engine 340. In another embodiment of this method, event collection database 320 is a logging system of a security device that generates events.

[0054] Policy configuration information 330 includes but is not limited to a global threat assessment event generation probability and one or more DTA rules containing event probability information. In another embodiment of this method, policy

configuration information 330 includes but is not limited to rule probability thresholds. In another embodiment of this method, policy configuration information 330 includes but is not limited to event collection database configurations. In another embodiment of this method, policy configuration information 330 includes but is not limited to operation parameters.

[0055] DTA engine 340 performs a number of functions. It accepts policy configuration information 330 and generates one or more abstract data types for one or more DTA rules. It reads each event in the event collection database 320. It determines if each event is a member of each instance of one or more abstract data types for each of one or more DTA rules. If the event is a member of the instance, it adds each event to an instance and computes a probability of the instance. It determines if the probability of the instance is greater than the global threat assessment event generation probability. If the probability of the instance is greater than the global threat assessment event generation probability, it generates a DTA event and places the DTA event in event collection database 320. It determines if the event is a starting member of an instance of one or more abstract data types for each of one or more DTA rules. Finally, if the event is a starting member of the instance, it creates the instance and adds the event to the instance.

[0056] In another embodiment of this system, DTA engine 340 determines if the probability of the instance is greater than a rule probability threshold for each instance.

[0057] In another embodiment of this system, DTA engine 340 removes an instance from memory, if the probability of the instance is greater than the global threat assessment event probability.

[0058] In another embodiment of this system, DTA engine 340 removes an instance from memory, if the probability of the instance is greater than the rule probability threshold of the instance.

[0059] In another embodiment of this system, a system management console includes event collection database 320, policy configuration information 330, and DTA engine 340.

[0060] Figure 4 is a flowchart showing the steps of an exemplary method for assessing a threat probability of an event generated by a security device in accordance with an embodiment of the present invention.

[0061] In step 410 of method 400, the event is received from a security device by an event collection database.

[0062] In step 420, if the event matches an unpopulated member of an instance of an abstract data type, the event is added to the instance and a probability of the instance is computed. The instance of the abstract data type represents a rule that describes how events are combined to form a threat.

[0063] In step 430, if the probability of the instance is greater than a global threat assessment event generation probability, a second event is generated and the second event is placed in the event collection database. The second event is a DTA event.

[0064] Method 400 describes the computation of a threat probability of an event by combining the probability of that event with the probabilities of other events that have

occurred. The threat is assessed by comparing the combined probability with a probability threshold. Method 400 also provides a feedback mechanism that places DTA events in the same event collection database as events generated by a security device. In another embodiment this method, this feedback mechanism is used to assess threats that are composed of other threats. In this embodiment, DTA events are obtained from the event collection database and are processed in the same manner as events generated from a security device.

[0065] The present invention is an important enhancement to event correlation and provides a powerful tool to mitigate attacks in real-time. The present invention can be used to detect more general types of approaches that might, for example, use port scanning, followed by service identification, and forecast a possible attack in progress even though the exact attack may be unknown or undetectable. This gives the analyst the ability to detect zero-day exploits given that the exploit follows previous attack methodologies. Zero-day exploits are attacks that have not yet been published, so no specific rules yet exist to protect against them.

[0066] In accordance with an embodiment of the present invention, instructions adapted to be executed by a processor to perform a method are stored on a computer-readable medium. The computer-readable medium can be a device that stores digital information. For example, a computer-readable medium includes a read-only memory (e.g., a Compact Disc-ROM ("CD-ROM") as is known in the art for storing software. The computer-readable medium can be accessed by a processor suitable for executing instructions adapted to be executed. The terms "instructions configured to be executed" and "instructions to be executed" are meant to encompass any

instructions that are ready to be executed in their present form (e.g., machine code) by a processor, or require further manipulation (e.g., compilation, decryption, or provided with an access code, etc.) to be ready to be executed by a processor.

[0067] In this detailed description, systems and methods in accordance with embodiments of the present invention have been described with reference to specific exemplary embodiments. Accordingly, the present description and figures are to be regarded as illustrative rather than restrictive.

[0068] Embodiments of the present invention relate to data communications via one or more networks. The data communications can be carried by one or more communications channels of the one or more networks. A network can include wired communication links (e.g., coaxial cable, copper wires, optical fibers, a combination thereof, and so on), wireless communication links (e.g., satellite communication links, terrestrial wireless communication links, satellite-to-terrestrial communication links, a combination thereof, and so on), or a combination thereof. A communications link can include one or more communications channels, where a communications channel carries communications. For example, a communications link can include multiplexed communications channels, such as time division multiplexing ("TDM") channels, frequency division multiplexing ("FDM") channels, code division multiplexing ("CDM") channels, wave division multiplexing ("WDM") channels, a combination thereof, and so on.

[0069] Systems and methods in accordance with an embodiment of the present invention disclosed herein can advantageously reduce the number of successful

computer intrusions by identifying collections of IDS events as computer or network attacks.

[0070] In the foregoing detailed description, systems and methods in accordance with embodiments of the present invention have been described with reference to specific exemplary embodiments. Accordingly, the present specification and figures are to be regarded as illustrative rather than restrictive. The scope of the invention is to be further understood by the numbered examples appended hereto, and by their equivalents.